Docket No. 50277-1774

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

Rae K. Burns, et al.

Serial No.: 10/006,543

Filed: November 30, 2001

For: TECHNIQUES FOR ADDING MULTIPLE SECURITY POLICIES TO A DATABASE SYSTEM

Confirmation No.: 1004

Group Art Unit No.: 2164

Examiner: Wong, Leslie

Declaration Under 37 CFR 1.131

Sir:

We, RAE K. BURNS, and PATRICK F. SACK, and VIKRAM REDDY PESATI, pursuant to 37 CFR 1.131, declare:

1. We are the inventors named in the above referenced patent application ("Application").

2. We make this declaration for the purpose of establishing a reduction to practice of the inventions disclosed and claimed in the Application at a date prior to March 30, 2001, the effective filing date of U.S. Patent Application Publication No. US 2002/0143735, herein Ayi.

3. We conceived and reduced to practice an implementation of claims 1 – 5 and 21 - 25 before the effective filing date of Ayi.

4. We participated on a team that developed the implementation of claims 1 – 5 and 21 – 25 that is incorporated into an Oracle™ database server product. After the design phase of the development, successful tests were run to show that the implementation worked

according to claims 1 – 5 and 21 - 25. These tests, which were conducted using standard internal test processes and procedures, were completed before the effective filing date of Ayi and were carried out in this country.

5. Attached as Exhibit A is a true and correct print out of substantially all of test script file 'tzlas01.sql'. The test script was used to test the implementation.

6. Attached as Exhibit B is a true and correct printout of test script log file 'tzllas01.log', which shows the results of running the test script shown in Exhibit A before the effective filing date of Ayi. The results show that the tests were successful.

7. Attached as Exhibit C is a true and correct print out of substantially all of test script file 'tzlbac14.sql'. The test script was used to test the implementation.

8. Attached as Exhibit D is a true and correct printout of test script log file 'tzlbac14.log', which shows the results of running the test script shown in Exhibit C before the effective filing date of Ayi. The results show that the tests were successful.

9. Exhibit D has been annotated with bolded and bracketed comments that illustrate how Exhibit D supports the claim language of Claims 1-5 and 21-25, as required by the Examiner in the Office Action dated March 3, 2006.

10. Exhibits A, B, C, and D are submitted as probative of the fact that the successful tests referred to in paragraph 4 were executed before the filing date of Ayi.

Each person signing below states that all statements made herein of his own knowledge are true and that all statements made herein on information and belief are believed to be true, and further, that the statements are made with the knowledge that willful false statements in the like so made are punishable by a fine or imprisonment or both, under Section 101, Title 18 of the United States Code and that such willful and false statements may jeopardize the validity of the application or any patent issued thereon.

Dated: _____

RAE K. BURNS

Dated: _____

PATRICK F. SACK

Dated: MAY 03, 2006

P. Vikram Reddy

VIKRAM REDDY PESATI

Docket No. 50277-1774

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

Rae K. Burns, et al.

Serial No.: 10/006,543

Filed: November 30, 2001

For: TECHNIQUES FOR ADDING MULTIPLE SECURITY POLICIES TO A DATABASE SYSTEM

Confirmation No.: 1004

Group Art Unit No.: 2164

Examiner: Wong, Leslie

Declaration Under 37 CFR 1.131

Sir:

We, RAE K. BURNS, and PATRICK F. SACK, and VIKRAM REDDY PESATI, pursuant to 37 CFR 1.131, declare:

1. We are the inventors named in the above referenced patent application ("Application").

2. We make this declaration for the purpose of establishing a reduction to practice of the inventions disclosed and claimed in the Application at a date prior to March 30, 2001, the effective filing date of U.S. Patent Application Publication No. US 2002/0143735, herein Ayi.

3. We conceived and reduced to practice an implementation of claims 1 – 5 and 21 - 25 before the effective filing date of Ayi.

4. We participated on a team that developed the implementation of claims 1 – 5 and 21 – 25 that is incorporated into an Oracle™ database server product. After the design phase of the development, successful tests were run to show that the implementation worked

according to claims 1 – 5 and 21 - 25. These tests, which were conducted using standard internal test processes and procedures, were completed before the effective filing date of Ayi and were carried out in this country.

5. Attached as Exhibit A is a true and correct print out of substantially all of test script file 'tzlas01.sql'. The test script was used to test the implementation.

6. Attached as Exhibit B is a true and correct printout of test script log file 'tzllas01.log', which shows the results of running the test script shown in Exhibit A before the effective filing date of Ayi. The results show that the tests were successful.

7. Attached as Exhibit C is a true and correct print out of substantially all of test script file 'tzlbac14.sql'. The test script was used to test the implementation.

8. Attached as Exhibit D is a true and correct printout of test script log file 'tzlbac14.log', which shows the results of running the test script shown in Exhibit C before the effective filing date of Ayi. The results show that the tests were successful.

9. Exhibit D has been annotated with bolded and bracketed comments that illustrate how Exhibit D supports the claim language of Claims 1-5 and 21-25, as required by the Examiner in the Office Action dated March 3, 2006.

10. Exhibits A, B, C, and D are submitted as probative of the fact that the successful tests referred to in paragraph 4 were executed before the filing date of Ayi.

Each person signing below states that all statements made herein of his own knowledge are true and that all statements made herein on information and belief are believed to be true, and further, that the statements are made with the knowledge that willful false statements in the like so made are punishable by a fine or imprisonment or both, under Section 101, Title 18 of the United States Code and that such willful and false statements may jeopardize the validity of the application or any patent issued thereon.

Dated: May 4, 2006

RAE K. BURNS

Dated: _____

PATRICK F. SACK

Dated: _____

VIKRAM REDDY PESATI

Docket No. 50277-1774

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:

Rae K. Burns, et al.

Serial No.: 10/006,543

Filed: November 30, 2001

For: TECHNIQUES FOR ADDING MULTIPLE SECURITY POLICIES TO A DATABASE SYSTEM

Confirmation No.: 1004

Group Art Unit No.: 2164

Examiner: Wong, Leslie

Declaration Under 37 CFR 1.131

Sir:

We, RAE K. BURNS, and PATRICK F. SACK, and VIKRAM REDDY PESATI, pursuant to 37 CFR 1.131, declare:

1. We are the inventors named in the above referenced patent application ("Application").

2. We make this declaration for the purpose of establishing a reduction to practice of the inventions disclosed and claimed in the Application at a date prior to March 30, 2001, the effective filing date of U.S. Patent Application Publication No. US 2002/0143735, herein Ayi.

3. We conceived and reduced to practice an implementation of claims 1 - 5 and 21 - 25 before the effective filing date of Ayi.

4. We participated on a team that developed the implementation of claims 1 – 5 and 21 – 25 that is incorporated into an Oracle™ database server product. After the design phase of the development, successful tests were run to show that the implementation worked

according to claims 1 – 5 and 21 - 25. These tests, which were conducted using standard internal test processes and procedures, were completed before the effective filing date of Ayi and were carried out in this country.

5. Attached as Exhibit A is a true and correct print out of substantially all of test script file 'tzlas01.sql'. The test script was used to test the implementation.

6. Attached as Exhibit B is a true and correct printout of test script log file 'tzllas01.log', which shows the results of running the test script shown in Exhibit A before the effective filing date of Ayi. The results show that the tests were successful.

7. Attached as Exhibit C is a true and correct print out of substantially all of test script file 'tzlbac14.sql'. The test script was used to test the implementation.

8. Attached as Exhibit D is a true and correct printout of test script log file 'tzlbac14.log', which shows the results of running the test script shown in Exhibit C before the effective filing date of Ayi. The results show that the tests were successful.

9. Exhibit D has been annotated with bolded and bracketed comments that illustrate how Exhibit D supports the claim language of Claims 1-5 and 21-25, as required by the Examiner in the Office Action dated March 3, 2006.

10. Exhibits A, B, C, and D are submitted as probative of the fact that the successful tests referred to in paragraph 4 were executed before the filing date of Ayi.

Each person signing below states that all statements made herein of his own knowledge are true and that all statements made herein on information and belief are believed to be true, and further, that the statements are made with the knowledge that willful false statements in the like so made are punishable by a fine or imprisonment or both, under Section 101, Title 18 of the United States Code and that such willful and false statements may jeopardize the validity of the application or any patent issued thereon.

Dated: _____

RAE K. BURNS

Dated: 5/3/2006

PATRICK F. SACK

Dated: _____

VIKRAM REDDY PESATI

Exhibit A
'tzlas01.sql'

```
--.
--  $Header: tzlas01.sql ...
.
.
.
--
REMARK >>>> Set System Variables For Current SQLPlus Session <<<<
SET FEEDBACK 1
SET NUMWIDTH 10
SET PAGESIZE 24
SET LINESIZE 80
SET TRIMSPOOL ON
SET TAB OFF
SET DEFINE '^'
SET ECHO ON

CONNECT LBACSYS/LBACSYS

-- Create two SA policies
EXECUTE SA_SYSDBA.CREATE_POLICY('SA1','SA1_COL','ALL_CONTROL');
EXECUTE SA_SYSDBA.CREATE_POLICY('SA2','SA2_COL','NO_CONTROL');

-- Initialize PUBLIC labels for them
EXECUTE SA_LABELS.CREATE_LEVEL('SA1',0,'PUBLIC','PUBLIC Level');
EXECUTE SA_LABELS.CREATE_LEVEL('SA2',0,'PUBLIC','PUBLIC Level');

EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1',10,'public');
EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa2',10,'public');

-- Setup some labels for policy SA1
EXECUTE SA_LABELS.CREATE_LEVEL('sa1',10,'c','confidential');
EXECUTE SA_LABELS.CREATE_LEVEL('sa1',20,'s','SECRET');
EXECUTE SA_LABELS.CREATE_LEVEL('sa1',30,'ts','Top Secret');

EXECUTE SA_LABELS.CREATE_COMPARTMENT ('sa1', 5, 'A', 'ALPHA');
EXECUTE SA_LABELS.CREATE_COMPARTMENT ('sa1', 10, 'b', 'beta');

EXECUTE SA_LABELS.CREATE_GROUP ('sa1', 5, 'G1','group 1');
EXECUTE SA_LABELS.CREATE_GROUP ('sa1', 51, 'G2','group 2','G1');
EXECUTE SA_LABELS.CREATE_GROUP ('sa1', 52, 'G3','group 3','G1');

EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 200,'c');
EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 225,'c:b,a');
EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1',210,'c:a');
EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1',205,'c::g2');
EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 300,'s');
EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 310,'s:a');

-- Generate some labels
SELECT LABEL_TO_CHAR(TO_SA_LABEL('sa1','c:a:g1')) FROM DUAL;
SELECT LABEL_TO_CHAR(TO_SA_LABEL('sa1','s:a,b')) FROM DUAL;
SELECT LABEL_TO_CHAR(TO_SA_LABEL('sa1','public:a:g1')) FROM DUAL;

COL POLICY_NAME FORMAT A15
COL LABEL FORMAT A20
SELECT * FROM DBA_SA_LABELS;
```

```
col labelvalue format a20
col policy_name format a10
SELECT * from dba_sa_labels;

-- Set user labels
EXECUTE SA_USER_ADMIN.SET_LEVELS('sa1','scott','s','c');
EXECUTE SA_USER_ADMIN.SET_COMPARTMENTS('sa1','scott','a,b');
EXECUTE SA_USER_ADMIN.SET_GROUPS('sa1','scott','G1');
SELECT * FROM dba_sa_user_levels ORDER BY policy_name, user_name;
SELECT * FROM dba_sa_user_compartments ORDER BY policy_name, user_name;
SELECT * fROM dba_sa_user_groups ORDER BY policy_name, user_name;

-- Look at session labels
CONNECT scott/tiger

create or replace FUNCTION get_list (pol IN VARCHAR2)
RETURN VARCHAR2 IS
  test_list lbacsys.lbac_label_list;
begin
  test_list:=lbac_session.effective_labels(pol);
  RETURN label_list_to_named_char(test_list,'effective');
END;
/

select get_list('sa1') from dual;
select get_list('sa2') from dual;
```

Exhibit B
'tzlas01.log'

```
SQL> @tzlas01
SQL>
SQL> CONNECT LBACSYS/LBACSYS
Connected.
SQL>
SQL> -- Create two SA policies
SQL> EXECUTE SA_SYSDBA.CREATE_POLICY('SA1','SA1_COL','ALL_CONTROL');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_SYSDBA.CREATE_POLICY('SA2','SA2_COL','NO_CONTROL');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Initialize PUBLIC labels for them
SQL> EXECUTE SA_LABELS.CREATE_LEVEL('SA1',0,'PUBLIC','PUBLIC Level');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABELS.CREATE_LEVEL('SA2',0,'PUBLIC','PUBLIC Level');

PL/SQL procedure successfully completed.

SQL>
SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1',10,'public');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa2',10,'public');
BEGIN SA_LABEL_ADMIN.CREATE_LABEL('sa2',10,'public'); END;

*
ERROR at line 1:
ORA-12432: LBAC error: Label with the given label_tag: 10 already exists
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC_LABEL_ADMIN", line 57
ORA-06512: at line 1


SQL>
SQL> -- Setup some labels for policy SA1
SQL> EXECUTE SA_LABELS.CREATE_LEVEL('sa1',10,'c','confidential');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABELS.CREATE_LEVEL('sa1',20,'s','SECRET');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABELS.CREATE_LEVEL('sa1',30,'ts','Top Secret');

PL/SQL procedure successfully completed.

SQL>
SQL> EXECUTE SA_LABELS.CREATE_COMPARTMENT ('sa1', 5, 'A', 'ALPHA');
```

```
PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABELS.CREATE_COMPARTMENT ('sa1', 10, 'b', 'beta');

PL/SQL procedure successfully completed.

SQL>
SQL> EXECUTE SA_LABELS.CREATE_GROUP ('sa1', 5, 'G1','group 1');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABELS.CREATE_GROUP ('sa1', 51, 'G2','group 2','G1');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABELS.CREATE_GROUP ('sa1', 52, 'G3','group 3','G1');

PL/SQL procedure successfully completed.

SQL>
SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 200,'c');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 225,'c:b,a');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1',210,'c:a');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1',205,'c::g2');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 300,'s');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_LABEL_ADMIN.CREATE_LABEL('sa1', 310,'s:a');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Generate some labels
SQL> SELECT LABEL_TO_CHAR(TO_SA_LABEL('sa1','c:a:g1')) FROM DUAL;

LABEL_TO_CHAR(TO_SA_LABEL('SA1','C:A:G1'))
--------------------------------------------------------------------------------
-
C:A:G1

1 row selected.

SQL> SELECT LABEL_TO_CHAR(TO_SA_LABEL('sa1','s:a,b')) FROM DUAL;
```

```
LABEL_TO_CHAR(TO_SA_LABEL('SA1','S:A,B'))
--------------------------------------------------------------------------------
-
S:A,B

1 row selected.

SQL> SELECT LABEL_TO_CHAR(TO_SA_LABEL('sa1','public:a:g1')) FROM DUAL;

LABEL_TO_CHAR(TO_SA_LABEL('SA1','PUBLIC:A:G1'))
--------------------------------------------------------------------------------
-
PUBLIC:A:G1

1 row selected.

SQL>
SQL> COL POLICY_NAME FORMAT A15
SQL> COL LABEL FORMAT A20
SQL> SELECT * FROM DBA_SA_LABELS;

POLICY_NAME     LABEL                LABEL_TAG LABEL_TYPE
--------------- -------------------- ---------- ---------------
SA1             PUBLIC                       10 USER LABEL
SA1             C                           200 USER/DATA LABEL
SA1             C::G2                       205 USER/DATA LABEL
SA1             C:A                         210 USER/DATA LABEL
SA1             C:A,B                       225 USER/DATA LABEL
SA1             S                           300 USER/DATA LABEL
SA1             S:A                         310 USER/DATA LABEL
SA1             C:A:G1                1000000000 USER/DATA LABEL
SA1             S:A,B                 1000000001 USER/DATA LABEL
SA1             PUBLIC:A:G1           1000000002 USER/DATA LABEL

10 rows selected.

SQL>
SQL> col labelvalue format a20
SQL> col policy_name format a10
SQL> SELECT * from dba_sa_labels;

POLICY_NAM LABEL                LABEL_TAG LABEL_TYPE
---------- -------------------- ---------- ---------------
SA1        PUBLIC                       10 USER LABEL
SA1        C                           200 USER/DATA LABEL
SA1        C::G2                       205 USER/DATA LABEL
SA1        C:A                         210 USER/DATA LABEL
SA1        C:A,B                       225 USER/DATA LABEL
SA1        S                           300 USER/DATA LABEL
SA1        S:A                         310 USER/DATA LABEL
SA1        C:A:G1                1000000000 USER/DATA LABEL
SA1        S:A,B                 1000000001 USER/DATA LABEL
SA1        PUBLIC:A:G1           1000000002 USER/DATA LABEL

10 rows selected.

SQL>
```

```
SQL> -- Set user labels
SQL> EXECUTE SA_USER_ADMIN.SET_LEVELS('sa1','scott','s','c');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_USER_ADMIN.SET_COMPARTMENTS('sa1','scott','a,b');

PL/SQL procedure successfully completed.

SQL> EXECUTE SA_USER_ADMIN.SET_GROUPS('sa1','scott','G1');

PL/SQL procedure successfully completed.

SQL> SELECT * FROM dba_sa_user_levels ORDER BY policy_name, user_name;

POLICY_NAM USER_NAME                       MAX_LEVEL
---------- ------------------------------ ------------------------------
MIN_LEVEL                      DEF_LEVEL
------------------------------ ------------------------------
ROW_LEVEL
--------------------------------
SA1        SCOTT                           S
C                              S
S


1 row selected.

SQL> SELECT * FROM dba_sa_user_compartments ORDER BY policy_name, user_name;

POLICY_NAM USER_NAME                       COMP                           RW_AC
D
---------- ------------------------------ ------------------------------ -----
-
R
-
SA1        SCOTT                           A                              WRITE
Y
Y

SA1        SCOTT                           B                              WRITE
Y
Y


2 rows selected.

SQL> SELECT * fROM dba_sa_user_groups ORDER BY policy_name, user_name;

POLICY_NAM USER_NAME                       GRP                            RW_AC
D
---------- ------------------------------ ------------------------------ -----
-
R
-
SA1        SCOTT                           G1                             WRITE
Y
```

```
Y


1 row selected.

SQL>
SQL> -- Look at session labels
SQL> CONNECT scott/tiger
Connected.
SQL>
SQL> create or replace FUNCTION get_list (pol IN VARCHAR2)
  2  RETURN VARCHAR2 IS
  3    test_list lbacsys.lbac_label_list;
  4  begin
  5    test_list:=lbac_session.effective_labels(pol);
  6    RETURN label_list_to_named_char(test_list,'effective');
  7  END;
  8  /

Function created.

SQL>
SQL> select get_list('sa1') from dual;

GET_LIST('SA1')
--------------------------------------------------------------------------------
-
MAX READ LABEL='S:A,B:G1,G2,G3',MAX WRITE LABEL='S:A,B:G1,G2,G3',MIN WRITE
LABEL
='C',READ LABEL='S:A,B:G1,G2,G3',WRITE LABEL='S:A,B:G1,G2,G3',ROW
LABEL='S:A,B:G
1,G2,G3'


1 row selected.

SQL> select get_list('sa2') from dual;

GET_LIST('SA2')
--------------------------------------------------------------------------------
-


1 row selected.

SQL>
SQL> SQL>
```

Exhibit C
'tzlbac14.sql'

```
--.
--  $Header: tzlbac14.sql ...
.
.
.

REMARK >>>> Set System Variables For Current SQLPlus Session <<<<
SET FEEDBACK 1
SET NUMWIDTH 10
SET PAGESIZE 24
SET LINESIZE 80
SET TRIMSPOOL ON
SET TAB OFF
SET DEFINE '^'
SET ECHO ON

CONNECT SCOTT/TIGER

  CREATE TABLE abc
  (COL1 VARCHAR2(45));

  CREATE TABLE jing
  (COL1 VARCHAR2(45));

  GRANT ALL ON abc TO LBACSYS;

-- This should not be allowed as the user is SCOTT
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('simple','lbac$test1','raghu');

CONNECT LBACSYS/LBACSYS

-- Create policies in database

  EXECUTE LBAC_SYSDBA.CREATE_POLICY('simple','lbac$test1','raghu');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('complex','lbac$test1','rghu');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('sile','lbac$test1');

-- Error Conditions
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('complex','lbac$test1','aghu');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('dummy','lbac$test1');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('new','lbac$test1','raghu');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('new123','lbac$tt1','xyz');

-- This should not fail ...
  EXECUTE
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz1234','lbac$test1','fdf');

  EXECUTE
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz','lbac$test1','fdf');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('india','lbac$test1','fdfefg');

-- Add 5 policies due to max_label_policies default increase from 5 to 10.
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab1','lbac$test1','vc1');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab2','lbac$test1','vc2');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab3','lbac$test1','vc3');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab4','lbac$test1','vc4');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab5','lbac$test1','vc5');
```

```
-- Error Conditions
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('abc','lbac$test1','vcx');
  EXECUTE LBAC_SYSDBA.CREATE_POLICY('simple');
  EXECUTE
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz12345','lbac$test1','ragu'
);

-- Drop extra 5 policies from above.
  EXECUTE LBAC_SYSDBA.DROP_POLICY('ab1');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('ab2');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('ab3');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('ab4');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('ab5');

-- Let us check the policies created ...
  SELECT *
  FROM DBA_LBAC_POLICIES
  ORDER BY POLICY_NAME;

  EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple');

-- The basic objective from now on is to test the enable/disable procedures ...

  EXECUTE LBAC_LABEL_ADMIN.CREATE_LABEL('simple',1,'A,B', TRUE);
  EXECUTE LBAC_LABEL_ADMIN.CREATE_LABEL('abcdefghijklmnopqrstuvwxyz',2,'A',
TRUE);

  EXECUTE
LBAC_USER_ADMIN.SET_USER_LABELS('abcdefghijklmnopqrstuvwxyz','SCOTT',TO_LABEL_L
IST.FROM_CHAR('abcdefghijklmnopqrstuvwxyz',NULL,'A'));

  SELECT *
  FROM DBA_LBAC_USER_LABELS ORDER BY USER_NAME, POLICY_NAME;

-- Error Conditions
  EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('abcdefghijklmnopqrstuvwxyz','SCOTT','abc'
);

-- OK now
  EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('complex','SCOTT','abc','NO_CONTROL');
  EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('complex','SCOTT','jing','NO_CONTROL');
  EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('simple','SCOTT','abc','DELETE_CONTROL');
  EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('simple','SCOTT','EMP','DELETE_CONTROL');
  EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('sile','SCOTT','jing','DELETE_CONTROL');

CONNECT SCOTT/TIGER

  DESC abc;

  INSERT INTO abc(col1)
```

```
  VALUES('fdfd');

  UPDATE abc
  SET raghu = LBACSYS.TO_LBAC_LABEL('simple','A,B');

-- Should not allow ...
  DELETE FROM abc;

  SELECT col1,LABEL_TO_CHAR(raghu)
  FROM abc
  ORDER BY col1;

-- Error Condition
  EXECUTE LBAC_SYSDBA.DISABLE_POLICY('simple');

CONNECT LBACSYS/LBACSYS

  EXECUTE LBAC_SYSDBA.DISABLE_POLICY('simple');

-- Error Conditions ...
  EXECUTE LBAC_SYSDBA.DISABLE_POLICY('abcdefghijklmnopqrstuvwxyzfd');
  EXECUTE LBAC_SYSDBA.DISABLE_POLICY('abcdefghijklmnopqrstuvwxyz','fdf');

-- Should not delete as the disable will be effective from next session only
  DELETE FROM SCOTT.abc;

CONNECT SCOTT/TIGER

  SELECT col1,LABEL_TO_CHAR(raghu)
  FROM abc
  ORDER BY col1;

-- Should delete now as the policy is disabled ...
  DELETE FROM abc;

  SELECT col1,LABEL_TO_CHAR(raghu)
  FROM abc
  ORDER BY col1;

  INSERT INTO abc(col1)
  VALUES('123233');

-- Error Condition ...
  EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple');

CONNECT LBACSYS/LBACSYS

  EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple');

-- Error Conditions ...
  EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simpler1');
  EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple',FALSE);

-- Should delete now as the enable will be effective only from new session
  DELETE FROM SCOTT.abc;

CONNECT SCOTT/TIGER
```

```
-- Expecting no rows ...
  SELECT col1,LABEL_TO_CHAR(raghu)
  FROM abc
  ORDER BY col1;

  INSERT INTO abc(col1)
  VALUES('1232');

-- Delete should fail ...
  DELETE FROM abc;

  SELECT col1,LABEL_TO_CHAR(raghu)
  FROM abc
  ORDER BY col1;

CONNECT LBACSYS/LBACSYS

  EXECUTE LBAC_SYSDBA.DROP_POLICY('simple',TRUE);
  EXECUTE LBAC_SYSDBA.DROP_POLICY('complex', FALSE);
  EXECUTE LBAC_SYSDBA.DROP_POLICY('sile');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('abcdefghijklmnopqrstuvwxyz1234');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('abcdefghijklmnopqrstuvwxyz');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('india');

-- Error Conditions
  EXECUTE LBAC_SYSDBA.DROP_POLICY('adfd');
  EXECUTE LBAC_SYSDBA.DROP_POLICY('simple',XYZ);

  SELECT *
  FROM DBA_LBAC_POLICIES
  ORDER BY POLICY_NAME;

CONNECT SCOTT/TIGER

-- Simple policy was applied on two tables (abc,emp) the hidden column should
-- be dropped as the TRUE option is set; Policy complex was applied to
-- two tables(abc,jing) and hidden column should not be dropped as the option
-- was set to FALSE; Policy sile was applied to a table (jing) and the hidden
-- column should not be dropped as the default option is FALSE.

  DESC abc;
  DESC jing;
  DESC EMP;

  DROP TABLE abc;
  DROP TABLE jing;

SET ECHO OFF

EXIT;
```

Exhibit D
'tzlbac14.log'

```
SQL> @tzlbac14
SQL>
SQL> CONNECT SCOTT/TIGER
Connected.
SQL>
SQL>   CREATE TABLE abc
  2    (COL1 VARCHAR2(45));

Table created.

SQL>
SQL>   CREATE TABLE jing
  2    (COL1 VARCHAR2(45));

Table created.

SQL>
SQL>   GRANT ALL ON abc TO LBACSYS;

Grant succeeded.

SQL>
SQL> -- This should not be allowed as the user is SCOTT
SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('simple','lbac$test1','raghu');
BEGIN LBAC_SYSDBA.CREATE_POLICY('simple','lbac$test1','raghu'); END;

      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00201: identifier 'LBACSYS.LBAC_SYSDBA' must be declared
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored


SQL>
SQL> CONNECT LBACSYS/LBACSYS
Connected.
[START: CLAIMS 1 and 21
Shows a plurality of label-based policies that are created in the database; 3rd
parameter is the policy column name]
SQL>
SQL> -- Create policies in database
SQL>
SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('simple','lbac$test1','raghu');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('complex','lbac$test1','rghu');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('sile','lbac$test1');

PL/SQL procedure successfully completed.

[END: CLAIMS 1 and 21]
```

```
SQL>
SQL> -- Error Conditions
SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('complex','lbac$test1','aghu');
BEGIN LBAC_SYSDBA.CREATE_POLICY('complex','lbac$test1','aghu'); END;

*
ERROR at line 1:
ORA-12447: policy role already exists for policy complex
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-01921: role name 'COMPLEX_DBA' conflicts with another user or role name
ORA-06512: at line 1


SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('dummy','lbac$test1');
BEGIN LBAC_SYSDBA.CREATE_POLICY('dummy','lbac$test1'); END;

*
ERROR at line 1:
ORA-12442: policy column "TESTLABEL" already used by an existing policy
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line .*


SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('new','lbac$test1','raghu');
BEGIN LBAC_SYSDBA.CREATE_POLICY('new','lbac$test1','raghu'); END;

*
ERROR at line 1:
ORA-12442: policy column "RAGHU" already used by an existing policy
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line .*


SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('new123','lbac$tt1','xyz');
BEGIN LBAC_SYSDBA.CREATE_POLICY('new123','lbac$tt1','xyz'); END;

*
ERROR at line 1:
ORA-12412: policy package lbac$tt1 is not installed
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line .*


SQL>
SQL> -- This should not fail ...
SQL>   EXECUTE
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz1234','lbac$test1','fdf');

PL/SQL procedure successfully completed.

SQL>
SQL>   EXECUTE
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz','lbac$test1','fdf');
BEGIN
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz','lbac$test1','fdf'); END;
```

```
*
ERROR at line 1:
ORA-12447: policy role already exists for policy abcdefghijklmnopqrstuvwxyz
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-01921: role name 'ABCDEFGHIJKLMNOPQRSTUVWXYZ_DBA' conflicts with another
user or role name
ORA-06512: at line 1


SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('india','lbac$test1','fdfefg');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Add 5 policies due to max_label_policies default increase from 5 to 10.
SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab1','lbac$test1','vc1');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab2','lbac$test1','vc2');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab3','lbac$test1','vc3');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab4','lbac$test1','vc4');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('ab5','lbac$test1','vc5');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Error Conditions
SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('abc','lbac$test1','vcx');
BEGIN LBAC_SYSDBA.CREATE_POLICY('abc','lbac$test1','vcx'); END;

*
ERROR at line 1:
ORA-12422: max policies exceeded
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line 1


SQL>   EXECUTE LBAC_SYSDBA.CREATE_POLICY('simple');
BEGIN LBAC_SYSDBA.CREATE_POLICY('simple'); END;

      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00306: wrong number or types of arguments in call to 'CREATE_POLICY'
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored
```

```
SQL>   EXECUTE
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz12345','lbac$test1','ragu')
;
BEGIN
LBAC_SYSDBA.CREATE_POLICY('abcdefghijklmnopqrstuvwxyz12345','lbac$test1','ragu')
; END;

*
ERROR at line 1:
ORA-12447: policy role already exists for policy
abcdefghijklmnopqrstuvwxyz12345
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-01921: role name 'ABCDEFGHIJKLMNOPQRSTUVWXYZ_DBA' conflicts with another
user or role name
ORA-06512: at line 1


SQL>
SQL> -- Drop extra 5 policies from above.
SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('ab1');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('ab2');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('ab3');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('ab4');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('ab5');

PL/SQL procedure successfully completed.
```

[START: CLAIMS 1 and 21 - list of policies created]

```
SQL> -- Let us check the policies created ...
SQL>   SELECT *
  2    FROM DBA_LBAC_POLICIES
  3    ORDER BY POLICY_NAME;

POLICY_NAME                    COLUMN_NAME
------------------------------ ------------------------------
PACKAGE                          BIN_SIZE STATUS
------------------------------ ---------- --------
DEFAULT_FORMAT                 POLICY_FORMAT
------------------------------ ------------------------------
POLICY_OPTIONS
--------------------------------------------------------------------------------
```

```
DATABASE_LABELS
--------------------------------------------------------------------------------
ABCDEFGHIJKLMNOPQRSTUVWXYZ1234 FDF
LBAC$TEST1                              1 ENABLED


COMPLEX                        RGHU
LBAC$TEST1                              1 ENABLED


INDIA                          FDFEFG

POLICY_NAME                    COLUMN_NAME
------------------------------ ------------------------------
PACKAGE                          BIN_SIZE STATUS
------------------------------ ---------- --------
DEFAULT_FORMAT                 POLICY_FORMAT
------------------------------ ------------------------------
POLICY_OPTIONS
--------------------------------------------------------------------------------
DATABASE_LABELS
--------------------------------------------------------------------------------
LBAC$TEST1                              1 ENABLED



SILE                           TESTLABEL
LBAC$TEST1                              1 ENABLED



SIMPLE                         RAGHU
LBAC$TEST1                              1 ENABLED

POLICY_NAME                    COLUMN_NAME
------------------------------ ------------------------------
PACKAGE                          BIN_SIZE STATUS
------------------------------ ---------- --------
DEFAULT_FORMAT                 POLICY_FORMAT
------------------------------ ------------------------------
POLICY_OPTIONS
--------------------------------------------------------------------------------
DATABASE_LABELS
--------------------------------------------------------------------------------

5 rows selected.
```

[END: CLAIMS 1 and 21]

```
SQL>
SQL>   EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple');
```

[STEP 101: "Simple" policy is enabled; STEPS 101-104 illustrate how it is determined "whether to perform [an] operation on a row of a table based on a set of labels associated with the row, the set of labels corresponding to the policy set", as recited by Claims 1 and 21.]

```
PL/SQL procedure successfully completed.

SQL>
SQL> -- The basic objective from now on is to test the enable/disable procedures
...
SQL>
SQL>   EXECUTE LBAC_LABEL_ADMIN.CREATE_LABEL('simple',1,'A,B', TRUE);
[STEP 102: Label "A,B" is associated with policy "simple".]

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_LABEL_ADMIN.CREATE_LABEL('abcdefghijklmnopqrstuvwxyz',2,'A',
TRUE);
BEGIN LBAC_LABEL_ADMIN.CREATE_LABEL('abcdefghijklmnopqrstuvwxyz',2,'A', TRUE);
END;

*
ERROR at line 1:
ORA-12416: policy abcdefghijklmnopqrstuvwxyz not found
ORA-06512: at "LBACSYS.LBAC_CACHE", line .*
ORA-06512: at "LBACSYS.LBAC_LABEL_ADMIN", line .*
ORA-06512: at line 1

SQL>
SQL>   EXECUTE
LBAC_USER_ADMIN.SET_USER_LABELS('abcdefghijklmnopqrstuvwxyz','SCOTT',TO_LABEL_LI
ST.FROM_CHAR('abcdefghijklmnopqrstuvwxyz',NULL,'A'));
BEGIN
LBAC_USER_ADMIN.SET_USER_LABELS('abcdefghijklmnopqrstuvwxyz','SCOTT',TO_LABEL_LI
ST.FROM_CHAR('abcdefghijklmnopqrstuvwxyz',NULL,'A')); END;

*
ERROR at line 1:
ORA-01405: fetched column value is NULL

SQL>
SQL>   SELECT *
  2    FROM DBA_LBAC_USER_LABELS ORDER BY USER_NAME, POLICY_NAME;

no rows selected

SQL>
SQL> -- Error Conditions
SQL>   EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('abcdefghijklmnopqrstuvwxyz','SCOTT','abc')
;
BEGIN
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('abcdefghijklmnopqrstuvwxyz','SCOTT','abc')
; END;

*
ERROR at line 1:
ORA-12416: policy abcdefghijklmnopqrstuvwxyz not found
ORA-06512: at "LBACSYS.LBAC_CACHE", line .*
ORA-06512: at "LBACSYS.LBAC_POLICY_ADMIN", line .*
ORA-06512: at line 1
```

[START: CLAIMS 1, 2, 5, 21, 22, and 25
For CLAIMS 1 and 21, policy "complex" is applied to jing and policies "complex" and "simple" are applied to table "abc".
For CLAIMS 2 and 22, a policy column is added when a policy is applied to a table (e.g. "abc"). (See next bolded section on this page).
For CLAIMS 5 and 25, policy "complex" and "simple" are the two or more policies of the plurality of label-based policies of table "abc"]

```
SQL>
SQL> -- OK now
SQL>   EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('complex','SCOTT','abc','NO_CONTROL');

PL/SQL procedure successfully completed.

SQL>   EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('complex','SCOTT','jing','NO_CONTROL');

PL/SQL procedure successfully completed.

SQL>   EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('simple','SCOTT','abc','DELETE_CONTROL');

PL/SQL procedure successfully completed.
```

[END: CLAIMS 1, 2, 5, 21, 22, and 25]

```
SQL>   EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('simple','SCOTT','EMP','DELETE_CONTROL');

PL/SQL procedure successfully completed.

SQL>   EXECUTE
LBAC_POLICY_ADMIN.APPLY_TABLE_POLICY('sile','SCOTT','jing','DELETE_CONTROL');

PL/SQL procedure successfully completed.

SQL>
SQL> CONNECT SCOTT/TIGER
Connected.
```

[Step 103: User 'SCOTT' connects to the database. Note: user 'SCOTT is not associated with any labels.]

[START: CLAIM 2, 5, 22 and 25
For CLAIMS 2 and 22, policy columns "RGHU" and "RAGHU" are added for policies "complex" and "simple".
For CLAIMS 5 and 25, these columns are added because policies "complex" and "simple" are applied - i.e. "the policy set associated with the table includes two or more policies of the plurality of label-based policies."]

```
SQL>
SQL>   DESC abc;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 COL1                                               VARCHAR2(45)
 RGHU                                               LBACSYS.LBAC_LABEL
 RAGHU                                              LBACSYS.LBAC_LABEL
```

[END: CLAIM 2, 5, 22 and 25]

```
SQL>
SQL>    INSERT INTO abc(col1)
  2     VALUES('fdfd');

1 row created.
```

[START: CLAIMS 3 and 23
Label "A,B" for policy "simple" is stored in policy column "raghu" corresponding to the policy.]

```
SQL>
SQL>    UPDATE abc
  2     SET raghu = LBACSYS.TO_LBAC_LABEL('simple','A,B');

1 row updated.
```

[END: CLAIMS 3 and 23]

[Step 104/START: CLAIMS 1, 4, 21 and 24
For CLAIMS 1 and 21, the following delete operation is received. It was previously shown which policies apply to table "abc". Here, only policy "simple" is enforced on delete because DELETE_CONTROL is only specified for policy "simple" even though both policies "simple" and "complex" are applied. Thus, policy "simple", of the plurality of label-based policies ("simple" and "complex"), was determined to be applied to table "abc". In this example, it is determined that the delete operation is NOT performed based on the set of labels associated with the row (i.e. user "SCOTT" is not associated with any labels so the user is denied the ability to delete the row).
For CLAIMS 4 and 24, in order to determine which policies apply, it must be determined whether a column is a policy column]

```
SQL> -- Should not allow ...
SQL>    DELETE FROM abc;
  DELETE FROM abc
              *
ERROR at line 1:
ORA-12406: unauthorized SQL statement for policy SIMPLE
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC.*
ORA-04088: error during execution of trigger 'LBACSYS.LBAC.*
```

[END: CLAIMS 1, 4, 21, and 24]

[START: CLAIMS 3 and 23
Shows that label "A,B" associated with policy "simple" is stored in policy column "raghu" of a row, in table "abc", with value "fdfd"]

```
SQL>
SQL>    SELECT col1,LABEL_TO_CHAR(raghu)
  2     FROM abc
  3     ORDER BY col1;

COL1
--------------------------------------------
LABEL_TO_CHAR(RAGHU)
--------------------------------------------------------------------------------
fdfd
A,B

1 row selected.
```

[END: CLAIMS 3 and 23]

```
SQL>
SQL> -- Error Condition
SQL>   EXECUTE LBAC_SYSDBA.DISABLE_POLICY('simple');
BEGIN LBAC_SYSDBA.DISABLE_POLICY('simple'); END;

      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00201: identifier 'LBACSYS.LBAC_SYSDBA' must be declared
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored


SQL>
SQL> CONNECT LBACSYS/LBACSYS
Connected.
SQL>
SQL>   EXECUTE LBAC_SYSDBA.DISABLE_POLICY('simple');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Error Conditions ...
SQL>   EXECUTE LBAC_SYSDBA.DISABLE_POLICY('abcdefghijklmnopqrstuvwxyzfd');
BEGIN LBAC_SYSDBA.DISABLE_POLICY('abcdefghijklmnopqrstuvwxyzfd'); END;

*
ERROR at line 1:
ORA-12416: policy abcdefghijklmnopqrstuvwxyzfd not found
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line 1


SQL>   EXECUTE LBAC_SYSDBA.DISABLE_POLICY('abcdefghijklmnopqrstuvwxyz','fdf');
BEGIN LBAC_SYSDBA.DISABLE_POLICY('abcdefghijklmnopqrstuvwxyz','fdf'); END;

      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00306: wrong number or types of arguments in call to 'DISABLE_POLICY'
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored

SQL>
SQL> -- Should not delete as the disable will be effective from next session
only
SQL>   DELETE FROM SCOTT.abc;

1 row deleted.

SQL>
SQL> CONNECT SCOTT/TIGER
Connected.
SQL>
SQL>   SELECT col1,LABEL_TO_CHAR(raghu)
  2    FROM abc
  3    ORDER BY col1;
```

```
no rows selected

SQL>
SQL> -- Should delete now as the policy is disabled ...
SQL>   DELETE FROM abc;

0 rows deleted.

SQL>
SQL>   SELECT col1,LABEL_TO_CHAR(raghu)
  2    FROM abc
  3    ORDER BY col1;

no rows selected

SQL>
SQL>   INSERT INTO abc(col1)
  2    VALUES('123233');

1 row created.

SQL>
SQL> -- Error Condition ...
SQL>   EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple');
BEGIN LBAC_SYSDBA.ENABLE_POLICY('simple'); END;

      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00201: identifier 'LBACSYS.LBAC_SYSDBA' must be declared
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored


SQL>
SQL> CONNECT LBACSYS/LBACSYS
Connected.
SQL>
SQL>   EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Error Conditions ...
SQL>   EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simpler1');
BEGIN LBAC_SYSDBA.ENABLE_POLICY('simpler1'); END;

*
ERROR at line 1:
ORA-12416: policy simpler1 not found
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line 1


SQL>   EXECUTE LBAC_SYSDBA.ENABLE_POLICY('simple',FALSE);
BEGIN LBAC_SYSDBA.ENABLE_POLICY('simple',FALSE); END;
```

```
      *
ERROR at line 1:
ORA-06550: line 1, column 7:
PLS-00306: wrong number or types of arguments in call to 'ENABLE_POLICY'
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored

SQL>
SQL> -- Should delete now as the enable will be effective only from new session
SQL>   DELETE FROM SCOTT.abc;

1 row deleted.

SQL>
SQL> CONNECT SCOTT/TIGER
Connected.
SQL>
SQL> -- Expecting no rows ...
SQL>   SELECT col1,LABEL_TO_CHAR(raghu)
  2    FROM abc
  3    ORDER BY col1;

no rows selected

SQL>
SQL>   INSERT INTO abc(col1)
  2    VALUES('1232');

1 row created.

SQL>
SQL> -- Delete should fail ...
SQL>   DELETE FROM abc;
  DELETE FROM abc
              *
ERROR at line 1:
ORA-12406: unauthorized SQL statement for policy SIMPLE
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC.*
ORA-04088: error during execution of trigger 'LBACSYS.LBAC.*


SQL>
SQL>   SELECT col1,LABEL_TO_CHAR(raghu)
  2    FROM abc
  3    ORDER BY col1;

COL1
----------------------------------------
LABEL_TO_CHAR(RAGHU)
--------------------------------------------------------------------------------
1232


1 row selected.
```

```
SQL>
SQL> CONNECT LBACSYS/LBACSYS
Connected.
SQL>
SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('simple',TRUE);

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('complex', FALSE);

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('sile');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('abcdefghijklmnopqrstuvwxyz1234');

PL/SQL procedure successfully completed.

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('abcdefghijklmnopqrstuvwxyz');
BEGIN LBAC_SYSDBA.DROP_POLICY('abcdefghijklmnopqrstuvwxyz'); END;

*
ERROR at line 1:
ORA-12416: policy abcdefghijklmnopqrstuvwxyz not found
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line 1

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('india');

PL/SQL procedure successfully completed.

SQL>
SQL> -- Error Conditions
SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('adfd');
BEGIN LBAC_SYSDBA.DROP_POLICY('adfd'); END;

*
ERROR at line 1:
ORA-12416: policy adfd not found
ORA-06512: at "LBACSYS.LBAC_STANDARD", line 0
ORA-06512: at "LBACSYS.LBAC_SYSDBA", line .*
ORA-06512: at line 1

SQL>   EXECUTE LBAC_SYSDBA.DROP_POLICY('simple',XYZ);
BEGIN LBAC_SYSDBA.DROP_POLICY('simple',XYZ); END;
                                             *
ERROR at line 1:
ORA-06550: line 1, column 40:
PLS-00201: identifier 'XYZ' must be declared
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored


SQL>
```

```
SQL>   SELECT *
  2    FROM DBA_LBAC_POLICIES
  3    ORDER BY POLICY_NAME;

no rows selected

SQL>
SQL> CONNECT SCOTT/TIGER
Connected.
SQL>
SQL> -- Simple policy was applied on two tables (abc,emp) the hidden column
should
SQL> -- be dropped as the TRUE option is set; Policy complex was applied to
SQL> -- two tables(abc,jing) and hidden column should not be dropped as the
option
SQL> -- was set to FALSE; Policy sile was applied to a table (jing) and the
hidden
SQL> -- column should not be dropped as the default option is FALSE.
SQL>
SQL>   DESC abc;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 COL1                                               VARCHAR2(45)
 RGHU                                               LBACSYS.LBAC_LABEL

SQL>   DESC jing;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 COL1                                               VARCHAR2(45)
 RGHU                                               LBACSYS.LBAC_LABEL
 TESTLABEL                                          LBACSYS.LBAC_LABEL

SQL>   DESC EMP;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMPNO                                     NOT NULL NUMBER(4)
 ENAME                                              VARCHAR2(10)
 JOB                                                VARCHAR2(9)
 MGR                                                NUMBER(4)
 HIREDATE                                           DATE
 SAL                                                NUMBER(7,2)
 COMM                                               NUMBER(7,2)
 DEPTNO                                             NUMBER(2)

SQL>
SQL>   DROP TABLE abc;

Table dropped.

SQL>   DROP TABLE jing;

Table dropped.

SQL>
SQL> SET ECHO OFF
```